

Integrated Energy Management System

Adrian Vlose

1 Introducere

Proiectul se concentrează pe dezvoltarea unei aplicații care permite vizualizarea utilizatorilor și a dispozitivelor asociate fiecărui. Folosind un frontend realizat în React, împreună cu două microservicii dedicate pentru gestionarea utilizatorilor și a dispozitivelor, sistemul oferă o experiență interactivă. Backend-ul este construit cu Spring Boot, iar baza de date utilizată este MySQL.

2 Structura aplicației

- Backend-ul este format din 2 microservicii, unul pentru utilizatori și celălalt pentru dispozitive. Microserviciul utilizatorilor este format dintr-o aplicație Spring Boot și o bază de date stocată în MySQL. Cel de-al doilea microserviciu este implementat pe aceeași idee, dar are două tabele în loc de unul.
- Frontend-ul este realizat în React și este structurat în 3 pachete, câte unul pentru fiecare tip de utilizator, iar cel de-al treilea reprezintă pagina de home a interfetei.
- Toate acestea sunt salvate în Docker, prin intermediul Dockerfile-urilor și a unui fisier docker-compose. Rularea se face direct din consolă, iar testarea se realizează din interfața creată în React.

3 Implementare

1. Primul microserviciu este format din 7 pachete. Pachetul de 'model' definește datele. De asemenea, avem pachetele de 'repository', 'serviceInterface', 'service' și 'controller', care leagă funcționalitățile aplicației pe partea de utilizator. Ultimul pachet este folosit pentru a stoca datele primite prin endpointuri în DTO-uri.

Pe lângă acestea, există și partea de RestTemplate, care face posibilă legătura dintre cele două microservicii. Aceasta se referă la adăugarea ID-ului utilizatorului în cea de-a doua tabelă din microserviciul de dispozitive și la ștergerea utilizatorului, care implică ștergerea din baza de date a microserviciului și din dispozitivele asociate utilizatorului curent, precum și ID-ul din cea de-a doua tabelă.

2. Cel de-al doilea microserviciu este implementat pe aceeași structură, dar nu are partea de RestTemplate, iar fiecare pachet conține câte 2 clase, una pentru tabela de utilizatori și alta pentru tabela de dispozitive.
3. Partea de interfață este realizată în React și este formată din 3 pachete.
 - Unul pentru pagina de home, unde fiecare utilizator își acesează contul, iar după aceasta este redirectionat în componenta de utilizator sau de admin, în funcție de rolul acestuia din baza de date.

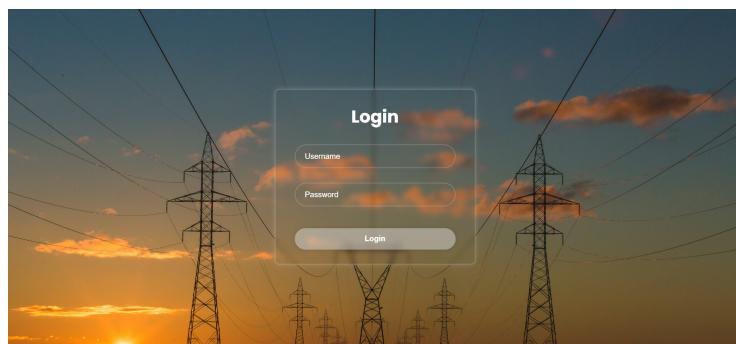


Figure 1: Pagina de home

- Pentru utilizator, acesta poate doar să vizualizeze lista de

dispozitive asociate lui.

Description	Address	Consumption
Ceiling Fan	Living Room	75
Temperature Sensor	123 Factory Lane	50
Pressure Valve	456 Industrial Blvd	75
Humidity Controller	789 Manufacturing Ave	30

Log out

Figure 2: Pagina de utilizator

- Pagina de admin, unde administratorul poate gestiona utilizatorii și dispozitivele. El poate modifica datele despre fiecare utilizator sau dispozitiv și poate adăuga unele noi.

Adrian		Show Users	Add User	Show Devices	Add Device	Log out	
Marian	Email: marian@gmail.com Password: marian	Edit	Delete	George	Email: george@gmail.com Password: george	Edit	Delete
Florin	Email: florin@gmail.com Password: florin	Edit	Delete	Ana	Email: ana@gmail.com Password: ana	Edit	Delete
		Back	Next				

Figure 3: Pagina de admin

4. În final, toate acestea au fost adăugate în Docker, prin intermediul a 3 fișiere Dockerfile pentru fiecare componentă, iar toate 3 au fost legate prin intermediul unui fișier docker-compose unde sunt precizate și serviciile bazei de date.

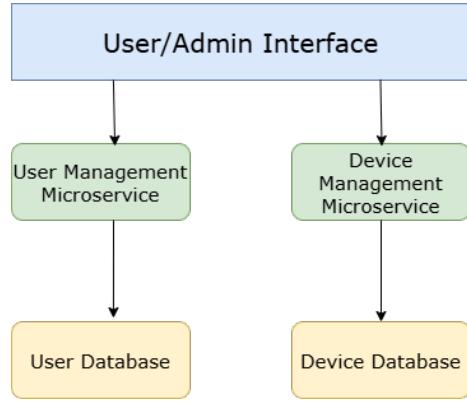


Figure 4: Diagrama arhitecturii conceptuale

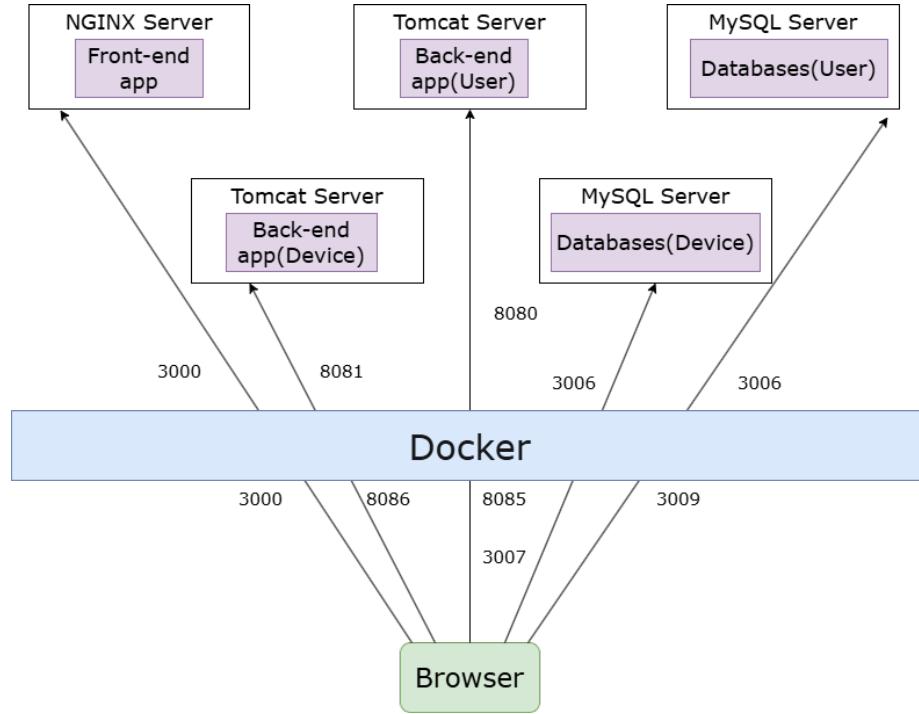


Figure 5: Diagrama de deployment

4 Funcționalități

4.1 Funcționalități interfață

4.1.1 Utilizator (User)

- Se poate loga în sistem.
- Poate vizualiza dispozitivele asociate lui.

4.1.2 Administrator (Admin)

- Se poate loga în sistem.
- Poate vizualiza toți utilizatorii.
- Poate modifica informațiile utilizatorilor existenți.
- Poate șterge utilizatori din sistem.
- Poate adăuga utilizatori noi în sistem.
- Poate vizualiza toate dispozitivele.
- Poate modifica informațiile despre dispozitivele existente.
- Poate șterge dispozitive din sistem.
- Poate adăuga dispozitive noi în sistem.

4.2 Endpointuri

4.2.1 Endpointuri : UserController

- **POST /user/insert:** Adaugă un utilizator nou în sistem.
- **DELETE /user/delete:** Șterge un utilizator din sistem.
- **POST /user/login:** Autentifică un utilizator.
- **POST /user/getID:** Obține ID-ul utilizatorului pe baza informațiilor furnizate.
- **POST /user/getUsers:** Obține toți utilizatorii din sistem.
- **PUT /user/update:** Actualizează informațiile unui utilizator existent.

4.2.2 Endpointuri : DeviceController

- **POST /device/insert:** Adaugă un dispozitiv nou în sistem.
- **POST /device/getByUser:** Obține toate dispozitivele asociate unui utilizator.
- **POST /device/getAll:** Obține toate dispozitivele din sistem.
- **PUT /device/update:** Actualizează informațiile unui dispozitiv existent.
- **DELETE /device/delete:** Sterge un dispozitiv din sistem.

5 Concluzie

In concluzie, aplicatia dezvoltata pentru gestionarea utilizatorilor si a dispozitelor ofera o interfata si functionalitati complete atat pentru utilizatori, cat si pentru administratori. Utilizatorii pot accesa usor dispozitivele asociate contului lor, iar administratorii au la dispozitie un set extins de functii pentru gestionarea utilizatorilor si a dispozitelor. Integrarea aplicatiei in Docker permite o implementare rapida si simpla, oferind in acelasi timp flexibilitate in gestionarea resurselor.